

Lecture 7: Sequential Networks

CK Cheng
Dept. of Computer Science and Engineering
University of California, San Diego

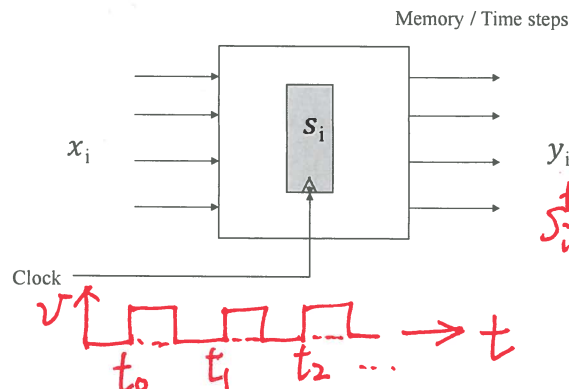
Transitions
States
History [Memory Timing] Future

1. Introduction: What is a sequential circuit?

“A circuit whose output depends on current inputs and past outputs”

“A circuit with **memory**”

Memory: a key parameter is Time



Combinational Logic

$$y_i = f_i(s^t, x)$$

$$s_i^{t+1} = g_i(s^t, x)$$

3

Part II: Sequential Networks

1. Introduction

1. Sequential circuits
2. Memory hierarchy
3. Basic mechanism of memory

2. Basic Building Blocks

1. Latches
2. Flip-Flops
3. Examples of Memory Modules

3. Implementation

2

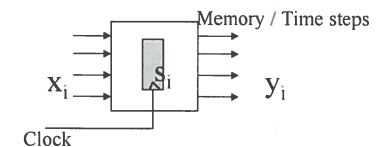
1. Introduction: Sequential Network Key features

Memory: Flip flops

Specification: Finite State Machines

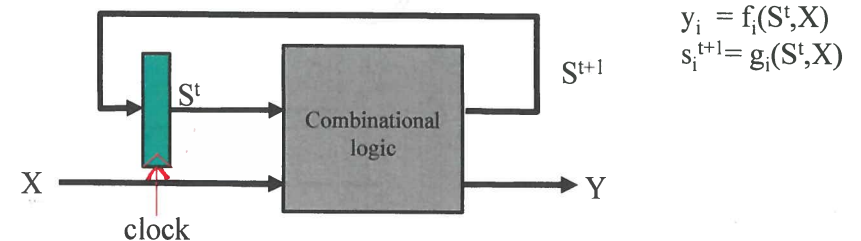
Implementation: Excitation Tables

Main Theme: Timing



Present time = t and **next time = t+1**

Timing constraints to separate the present and next times.



$$y_i = f_i(s^t, X)$$

$$s_i^{t+1} = g_i(s^t, X)$$

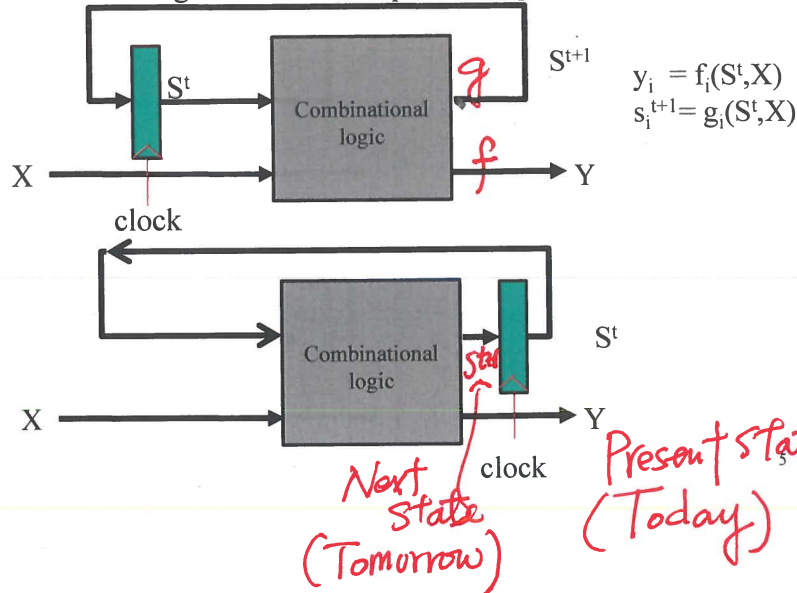
4

1. Introduction: Sequential Network Key features

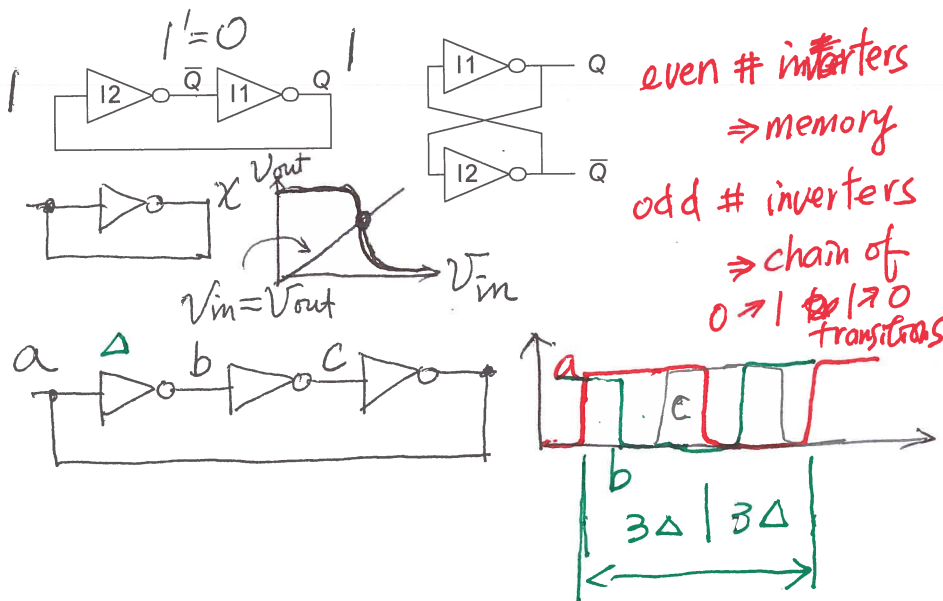
Main Theme: Timing

Present time = t and next time = $t+1$

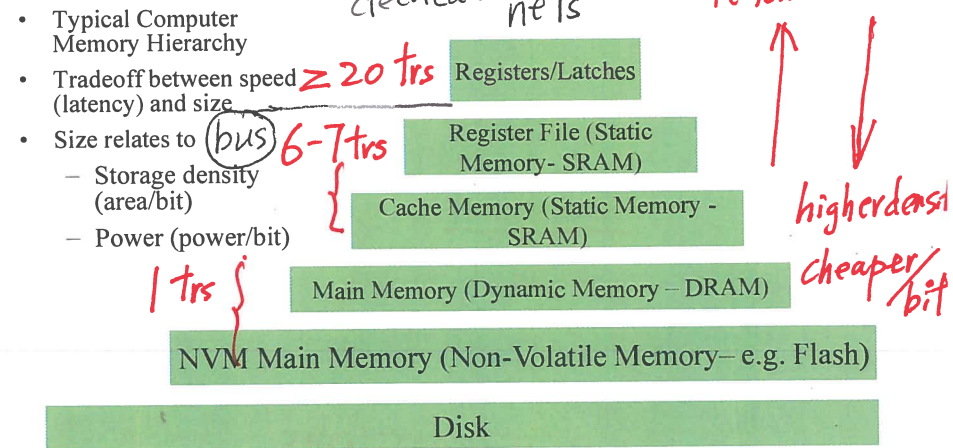
Timing constraints to separate the present and next times.



1. Introduction: Fundamental Memory Mechanism

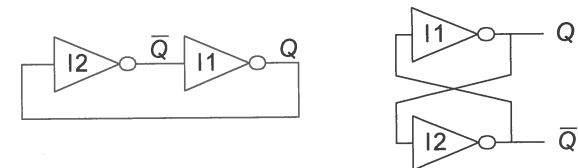


1. Introduction: Different Types of Memory/Storage



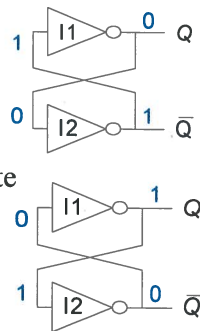
1. Introduction: Memory Mechanism Capacitive Load

- Fundamental building block of sequential circuits
- Two outputs: $\bar{Q}$, Q
- There is a feedback loop!
 - In a typical combinational logic, there is no feedback loop.
- No inputs



1. Introduction: Capacitive Loads

- Consider the two possible cases:
 - $Q = 0$: then $Q' = 1$ and $Q = 0$ (consistent)
 - $Q = 1$: then $Q' = 0$ and $Q = 1$ (consistent)
 - Bistable circuit stores 1 bit of state in the state variable, Q (or Q')
 - Hold the value due to capacitive charges and feedback loop strengthening
- But there are **no inputs to control the state**



9

Example

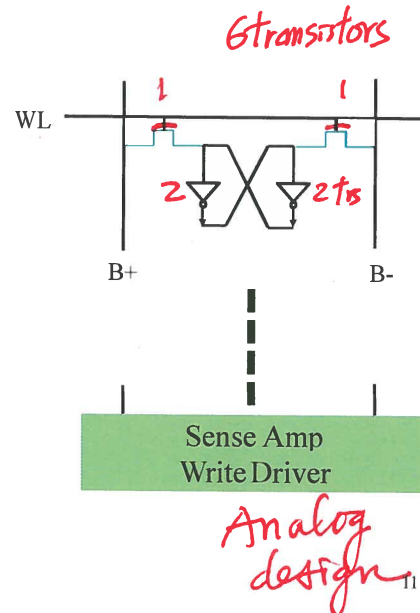
Q. Given a memory component made out of a loop of inverters, the number of inverters in the loop has to be

- A. Even
B. Odd
C. No constraints

10

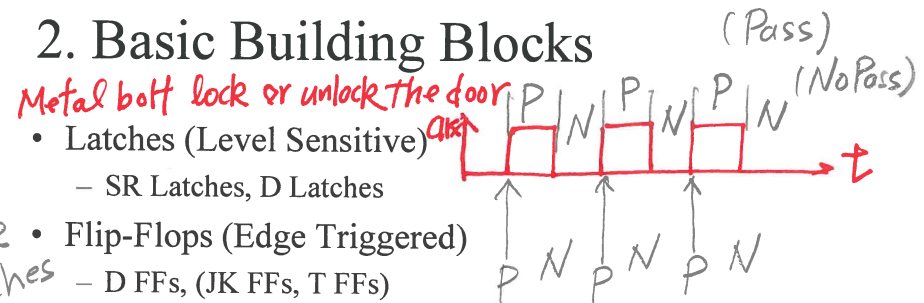
1. Introduction: Memory Storage Mechanism

- Word line (WL) to access cell
- Read – measure voltage difference between B+ and B-
- Write – force values of B+ and B- which may flip the cell
- Static Random Access Memory - SRAM



2. Basic Building Blocks

- Latches (Level Sensitive)
 - SR Latches, D Latches
- Flip-Flops (Edge Triggered)
 - D FFs, (JK FFs, T FFs)
- Examples of Memory Modules
 - Registers, Shift Registers, Pattern Recognizers, Counters, FIFOs

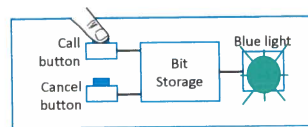


Double Latches

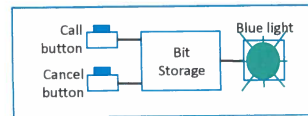
12

2. Basic Building Blocks: Flight attendant call button

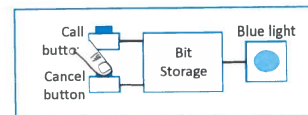
- Flight attendant call button
 - Press call: light turns on
 - Stays on** after button released
 - Press cancel: light turns off
 - Logic gate circuit to implement this?



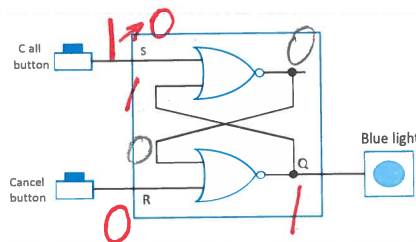
1. Call button pressed - light turns on



2. Call button released - light stays on



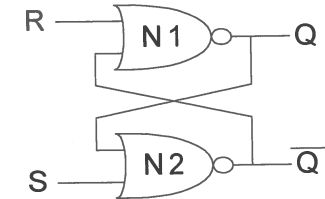
3. Cancel button pressed - light turns off



13

2. Basic Building Blocks: SR (Set/Reset) Latch

- SR Latch



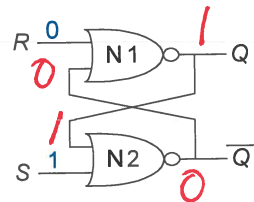
- Consider the four possible cases:

- $S = 1, R = 0$
- $S = 0, R = 1$
- $S = 0, R = 0$
- $S = 1, R = 1$

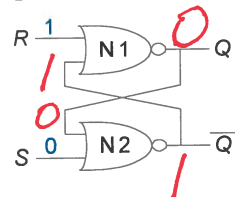
14

2. Basic Building Blocks: SR Latch Analysis

- $S = 1, R = 0$: then $Q = 1$ and $\bar{Q} = 0$



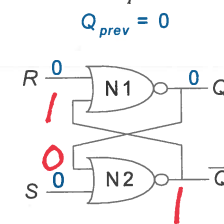
- $S = 0, R = 1$: then $Q = 0$ and $\bar{Q} = 1$



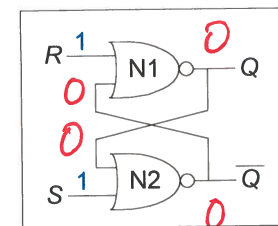
15

2. Basic Building Blocks: SR Latch Analysis

- $S = 0, R = 0$: then $Q = Q_{prev}$
 $Q_{prev} = 0$

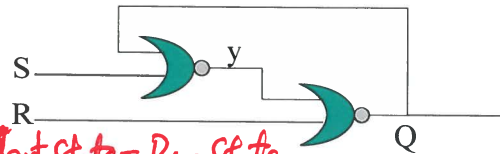
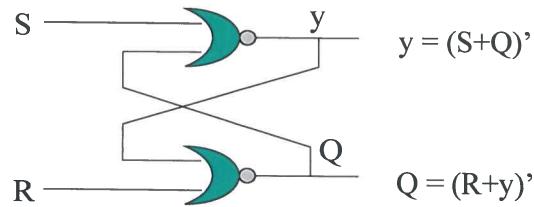


- $S = 1, R = 1$: then $Q = 0$ and $\bar{Q} = 0$



16

2. Basic Building Blocks: SR Latch



SR Q, y
 $00 \rightarrow y_{prev} \rightarrow \text{Next State} = \text{Pres. State}$
 $01 \rightarrow \text{Reset}$
 $10 \rightarrow \text{Set}$
 $11 \rightarrow 00$

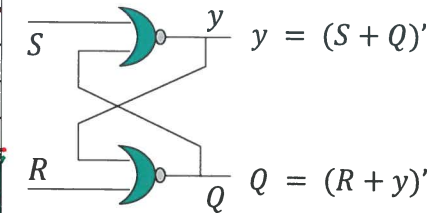
Inputs: S, R State: (Q, y)

17

id	S	R	Qt	yt	Qt	yt	Qt	yt	Qt	yt
0	0	0	0	0	1	1	0	0	1	1
1	0	0	0	1	0	1	0	1	0	1
2	0	0	1	0	1	0	1	0	1	0
3	0	0	1	1	0	0	1	1	0	0
4	0	1	0	0	0	1	0	1	0	1
5	0	1	0	1	0	1	0	1	0	1
6	0	1	1	0	0	0	0	1	0	1
7	0	1	1	1	0	0	0	1	0	1
8	1	0	0	0	1	0	1	0	1	0
9	1	0	0	1	0	0	1	0	1	0
10	1	0	1	0	1	0	1	0	1	0
11	1	0	1	1	0	0	1	0	1	0
12	1	1	0	0	0	0	0	0	0	0
13	1	1	0	1	0	0	0	0	0	0
14	1	1	1	0	0	0	0	0	0	0
15	1	1	1	1	0	0	0	0	0	0

Truth table of SR latch with incremental steps in time

Not state = Present state



18

id	S	R	Qt	yt	Qt	yt	Qt	yt	Qt	yt
0	0	0	0	0	1	1	0	0	1	1
1	0	0	0	1	0	1	0	1	0	1
2	0	0	1	0	1	0	1	0	1	0
3	0	0	1	1	0	0	1	1	0	0
4	0	1	0	0	0	1	0	1	0	1
5	0	1	0	1	0	1	0	1	0	1
6	0	1	1	0	0	0	0	1	0	1
7	0	1	1	1	0	0	0	1	0	1
8	1	0	0	0	1	0	1	0	1	0
9	1	0	0	1	0	0	1	0	1	0
10	1	0	1	0	1	0	1	0	1	0
11	1	0	1	1	0	0	1	0	1	0
12	1	1	0	0	0	0	0	0	0	0
13	1	1	0	1	0	0	0	0	0	0
14	1	1	1	0	0	0	0	0	0	0
15	1	1	1	1	0	0	0	0	0	0

"State Table" of SR latch

\SR	00	01	10	11
00	11	01	10	00
01	01	01	10	00
10	10	01	10	00
11	00	01	10	00

19

CASES:

SR=01: (Q,y) = (0,1)

SR=10: (Q,y) = (1,0)

SR=11: (Q,y) = (0,0)

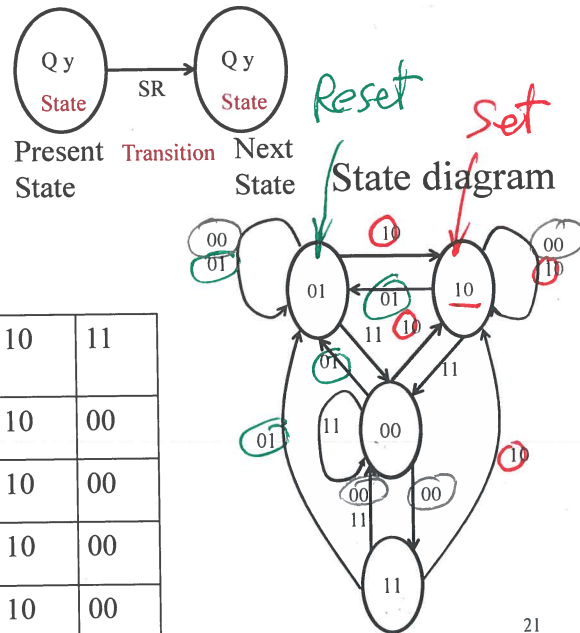
SR= 00: (Q,y) does not change if (Q,y)=(1,0) or (0,1)

However, when (Q,y) = (0,0) or (1,1), the output keeps changing

Remark: To verify the design, we need to enumerate all combinations.

20

State Table and State Diagram



State Table

\SR Qy	00	01	10	11
00	11	01	10	00
01	01	01	10	00
10	10	01	10	00
11	00	01	10	00

21

CASES:

SR=01: $(Q,y) = (0,1)$

SR=10: $(Q,y) = (1,0)$

SR=11: $(Q,y) = (0,0)$

SR= 00: (Q,y) does not change if $(Q,y)=(1,0)$ or $(0,1)$

However, when $(Q,y) = (0,0)$ or $(1,1)$, the output keeps changing

Q. Suppose that we can set the initial state $(Q,y)=(0,1)$. To avoid the SR latch output from toggling or behaving in an undefined way which input combinations should be avoided:

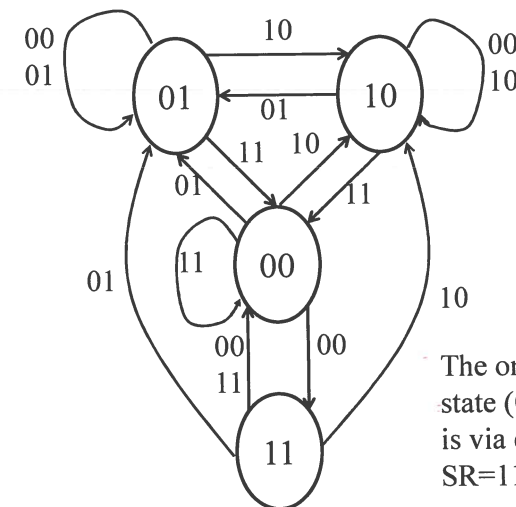
A. $(S, R) = (0, 0)$

B. $(S, R) = (1, 1)$

C. None of the above

22

SR Latch



The only way to reach state $(Q,y)=(0,0)$ or $(1,1)$ is via edge labeled SR=11.

CASES:

SR=01: $(Q,y) = (0,1)$

SR=10: $(Q,y) = (1,0)$

SR=11: $(Q,y) = (0,0)$

SR= 00: (Q,y) does not change if $(Q,y)=(1,0)$ or $(0,1)$

However, when $(Q,y) = (0,0)$ or $(1,1)$, the output keeps changing

We set the initial state $(Q,y)=(0,1)$ or $(1,0)$. To avoid the state $(Q,y)= (0,0)$ or $(1,1)$, we block the input SR=11.

Thus, without input SR=11, the state can only be $(Q,y)=(0,1)$ or $(1,0)$.

23

24